UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/700,152	11/03/2003	Agustin Gonzales-Tuchmann	5854-00200	3829

35617 7590 04/15/2009

DAFFER MCDANIEL LLP
P.O. BOX 684908
AUSTIN, TX 78768

EXAMINER	
KISS, ERIC B	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
04/15/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 10/700,152
Filing Date: November 03, 2003
Appellant(s): GONZALES-TUCHMANN ET AL.

Charles D. Huston (Reg. No. 31,027)
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed January 29, 2009, appealing from the Office action mailed January 10, 2008.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

No amendment after final has been filed.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is correct.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

6,874,141	SWAMY et al.	3-2005
6,947,947	BLOCK et al.	9-2005
6,449,619	COLLIAT et al.	9-2002

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claims 14-20 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Descriptive material can be characterized as either "functional descriptive material" or "nonfunctional descriptive material." In this context, "functional descriptive material" consists of data structures and computer programs which impart functionality when employed as a computer component. (The definition of "data structure" is "a physical or logical relationship among data elements, designed to support specific data manipulation functions." The New IEEE Standard Dictionary of Electrical and Electronics Terms 308 (5th ed. 1993).) "Nonfunctional descriptive material" includes but is not limited to music, literary works and a compilation or mere arrangement of data. Both types of "descriptive material" are nonstatutory when claimed as descriptive material *per se*. *In re Warmerdam*, 33 F.3d 1354, 1361, 31 USPQ2d 1754, 1760 (claim to a data structure per se held nonstatutory).

Data structures not claimed as embodied in computer-readable media are descriptive material *per se* and are not statutory because they are not capable of causing functional change in the computer. *See*, *e.g.*, *In re Warmerdam*, 33 F.3d 1354, 1361, 31 USPQ2d 1754, 1760 (claim to a data structure per se held nonstatutory). Such claimed data structures do not define any structural and functional interrelationships between the data structure and other claimed aspects of the invention which permit the data structure's functionality to be realized. In contrast, a claimed computer-readable medium encoded with a data structure defines structural and

functional interrelationships between the data structure and the computer software and hardware components which permit the data structure's functionality to be realized, and is thus statutory.

Similarly, computer programs claimed as computer listings per se, *i.e.*, the descriptions or expressions of the programs, are not physical "things." They are neither computer components nor statutory processes, as they are not "acts" being performed. Such claimed computer programs do not define any structural and functional interrelationships between the computer program and other claimed elements of a computer which permit the computer program's functionality to be realized. In contrast, a claimed computer-readable medium encoded with a computer program is a computer element which defines structural and functional interrelationships between the computer program and the rest of the computer which permit the computer program's functionality to be realized, and is thus statutory. *See In re Lowry*, 32 F.3d 1579, 1583-84, 32 USPQ2d 1031, 1035.

Claims 14-20 recite a "system" and an "environment" comprising a series of elements that can be reasonably interpreted as software, *per se*. The claims further recite a computer-readable medium "having" such elements. Claims 14 and 17 do not clearly and unambiguously recite the necessary functional and structural interrelationship between the software elements and the remaining elements of a computer. Specifically, the computer-readable medium "having" a particular environment does not necessarily imply storage of an executable program capable of causing a computer to carry out the described functionality. Accordingly, it is not clear that claims 14 and 17 recite more than functional descriptive material or whether such descriptive material is necessarily encoded in a manner sufficient to imply the necessary functional and structural interrelationship so to recite statutory subject matter.

Claims 1, 2, 4-14, and 16 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent No. 6,874,141 (Swamy et al.).

As per claim 1, *Swamy et al.* discloses:

developing one or more data transformations using a host language (see, e.g., col. 2, line 58, through col. 3, line 7, describing the overall schema mapping);

assembling several data transformations having ports into a map component with links between ports using a declarative language for static assemblage and a host language for dynamic assemblage (see, e.g., Figs. 11-14 (illustrating example mappings with links)), wherein a map component performs a respective data transformation (see, e.g., elements 356, 358, and 360 of Fig. 11, which transform input data into output data through multiplication and addition operations);

compiling one or more map components with syntactic and semantic analysis (see, e.g., Fig. 1 (describing the overall compilation process); col. 6, lines 34-39); and

synthesizing the compiled map components into an executable dataflow application including removing the design time links between ports (see, e.g., Fig. 1, describing the overall compilation process; Fig. 10 (freeing the node dependencies memory)).

As per claim 2, *Swamy et al.* further discloses the steps of creating an executable dataflow application comprising:

synthesizing a hierarchical map component (see, e.g., Fig. 4A; Fig. 11);

flattening the hierarchical map component (see, e.g., col. 10, lines 4-12; Fig. 5C (. . . compiler link option is flattening . . .)); and

determining the executable dataflow application using the flattened hierarchical map component in conjunction with runtime properties (see, e.g., col. 10, lines 4-12; Fig. 5C (flattening); Fig. 1 (describing the overall compilation process)).

As per claim 4, *Swamy et al.* further discloses the host language for data transformation logic being an object-oriented programming language (see, e.g., col. 7, lines 3-8).

As per claim 5, *Swamy et al.* further discloses some of the map components implementing dynamic assemblage logic implemented in an object-oriented programming language (see, e.g., col. 7, lines 3-8).

As per claim 6, *Swamy et al.* further discloses some of the map components comprising a plurality of other map components arranged hierarchically (see, e.g., Figs. 11-14 (illustrating example mappings with hierarchical components)).

As per claim 7, *Swamy et al.* further discloses some of the map components being static which consistently generate the same hierarchical map (see, e.g., Figs. 14 (illustrating an example mapping); col. 17, line 51, through col. 19, line 6 (illustrating direct copying of some mapped information from the source to the target)).

As per claim 8, *Swamy et al.* further discloses some of the map components being dynamic to generate different hierarchical maps dependent on properties and dynamic logic (see, e.g., col. 14, line 29, through col. Col. 15, lines 52 (describing the application of mathematical logic to produce results dependent on the properties of the source data)).

As per claim 9, *Swamy et al.* further discloses the dynamic map components receiving information concerning properties, port types, and dataflow graph implementation and

configuring its properties and internal dataflow graph implementation based on said received information (see, e.g., col. 15, lines 53-67).

As per claim 10, *Swamy et al.* further discloses one or more of the map components having interface and implementation properties (see, e.g., col. 15, lines 53-67).

As per claim 11, *Swamy et al.* further discloses some of the map components ports configured for sending data and some configured for receiving data (see, e.g., Fig. 11 (left and right side, respectively, of the individual mapping components)).

As per claim 12, *Swamy et al.* further discloses some of the ports being typed based on the type of data conveyed (see, e.g., Fig. 11 (the mathematical operations are designed to map numerical data)).

As per claim 13, *Swamy et al.* further discloses some of the ports being composite, comprising a plurality of hierarchical ports (see, e.g., Fig. 11 (multiple inputs/outputs to the mapping functions)).

As per claim 14, *Swamy et al.* discloses:

a library of components, some of the components being scalar and comprising a data transformation, some of the components being composite, and comprising a hierarchy of other components representing data transformations (see, e.g., Figs. 11-14 (illustrating example mappings with hierarchical components));

a compiler to develop and assemble components into an executable dataflow application linking components subject to schema and properties constraints (see, e.g., Fig. 1 (describing the overall compilation process); col. 6, lines 34-39; col. 2, lines 58-64 (“The invention takes a user-defined mapping between a source schema representing the definition of a source document, and

a target schema representing a target document, and compiles the mapping into code for execution in a computer system.”));

an executor which executes the dataflow application in parallel (see, e.g., col. 6, lines 34-39); and

tools for accessing the functionality of the compiler, executor and component library (see, e.g., Figs. 1, 2, and 15).

As per claim 16, *Swamy et al.* discloses some of the components including ports for accepting and producing data whereby the map components are linked (see, e.g., the illustrated mappings of Figs. 11-14; Fig. 11 (left and right side, respectively, of the individual mapping components)).

Claims 3 and 15 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,874,141 (Swamy et al.) in view of U.S. Patent No. 6,947,947 (Block et al.).

As per claims 3 and 15, although *Swamy et al.* fails to expressly disclose encrypting the dataflow graphs prior to the step of compiling the map component, *Block et al.* teaches the use of an encryption/decryption mechanism as part of a proprietary transformation (see, e.g., *Block et al.* at col. 6, line 44, through col. 7, line 13). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to utilize such encrypted proprietary transformations as a means of securing the transfer of transformed data over a network such as the Internet (see, e.g., *Block et al.* at col. 6, line 44, through col. 7, line 13).

Claims 17-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent No. 6,874,141 (Swamy et al.) in view of U.S. Patent No. 6,449,619 (Colliat et al.).

As per claim 17, *Swamy et al.* discloses:

A dataflow application development environment where map components are selected from a plurality of reusable map components each representing one or more data transformations (see, e.g., Figs. 11-14 (illustrating example mappings with hierarchical components)), the selected map components being visually assembled into a dataflow application (see, e.g., Figs. 11-14; col. 6, lines 34-39).

Swamy et al. further discloses at least some of the components being scalar components (see, e.g., *Swamy et al.* at col. 13, line 31, through col. 14, line 10), but fails to expressly disclose the recognition of patterns of parallelism and assigning of a number of threads to respective transformations and executing each scalar component on a separate thread. However, *Colliat et al.* teaches such multithreaded parallel transformation (see, e.g., *Colliat et al.* at col. 2, lines 43-62). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to utilize such parallel techniques as taught by *Colliat et al.* in order to gain the advantage of more efficient execution (see, e.g., *Colliat et al.* at col. 2, lines 43-62).

As per claim 18, *Swamy et al.* further discloses the map components defining properties affecting design behavior (see, e.g., col. 14, line 29, through col. Col. 15, lines 52 (describing the application of mathematical logic to produce results dependent on the properties of the source data)). Therefore, for reasons stated above, such a claim also would have been obvious.

As per claim 19, *Swamy et al.* further discloses the map components defining properties affecting execution behavior (see, e.g., col. 14, line 29, through col. Col. 15, lines 52 (describing the application of mathematical logic to produce results dependent on the properties of the source data)). Therefore, for reasons stated above, such a claim also would have been obvious.

As per claim 20, *Swamy et al.* further discloses the map components being assembled using ports for conveying data and declared as link points to other map components (see, e.g., Fig. 11 (left and right side, respectively, of the individual mapping components)). Therefore, for reasons stated above, such a claim also would have been obvious.

(10) Response to Argument

Rejection of claims 14-20 under 35 U.S.C. § 101 (Brief 4-6)

Appellants compare their claim 14 to claim 42 at issue in *Bilski*. (Brief 5.) The examiner agrees that both claims contain the word "having." However, there are significant differences which distinguish appellants' non-statutory claim 14 and the claim held to be a statutory *Beauregard* Claim in *Bilski*. Notably, the *Bilski* claim (as cited by appellants) recites, "having a computer readable program code embodied therein, said computer readable program code adapted to be executed to implement a method for generating a report." (Brief 5 (different portions emphasized by the examiner).) Such a claim clearly recites the necessary structural and functional interrelationship between the program code and the other elements of a computer such that the functionality of the program code can be realized as part of an operating machine. *See In re Lowry*, 32 F.3d 1579, 1583-84 (Fed. Cir. 1994). In contrast, appellants' claim 14 does not recite program code being actually stored on the medium (in fact, there is no discussion of such

storage in the originally filed specification, including the originally filed claims, to support such an interpretation).

Rejection of claims 1, 2, 4-14, and 16 under 35 U.S.C. § 102(e) (Brief 6-9)

Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

a) Appellants argue that, "Swamy does not perform port data type synthesis." (Brief 8.) However, this is not in the claim. To the extent that claim 1 requires "a map component performs a respective data transformation and "compiling one or more map components with syntactic and semantic analysis", Swamy meets these limitations. See, e.g., col. 6, lines 10-22 (describing the general compilation process, including processing of tree data structures representing the source schema and target schema in compiling a map defining data transformations).

b) Appellants argue that, "the compilation process used in Swamy does not produce an executable dataflow graph." (*Id.*) This is not precisely what is claimed. Instead, the claims recite generating an "executable dataflow application." See, e.g., Claim 1 (emphasis added). To the extent that Swamy discloses generating an application that performs data transformations on input data to produce output data, Swamy meets this limitation. See, e.g., Swamy at col. 2, lines 60-64 ("The invention takes a user-defined mapping between a source schema representing the definition of a source document, and a target schema representing a target document, and compiles the mapping into code for execution in a computer system.").

c) Appellants appear to argue that Swamy is limited to code generation and somehow does not disclose "developing one or more data transformations using a host language" and "assembling several data transformations having ports into a map component with links between ports using a declarative language for static assemblage and a host language for dynamic assemblage." (Brief 8.) However, Swamy does disclose such features. See, e.g., Swamy at Figs. 11-14 (illustrating example mappings with links); Fig. 11, elements 356, 358, and 360 (which transform input data into output data through multiplication and addition operations). The elements of the mappings, for example the illustrated multiplication and addition operations may be considered declarative in that they define rules for transformation of data from the source schema to the target schema. Further, the source schema and target schema may be considered a host language in that they represent the data to be dynamically transformed by the compiled mapping.

d) Appellants appear to argue that because some of the illustrated mappings of Swamy show transformation of scalar inputs to scalar outputs, that somehow Swamy does not disclose synthesizing the compiled map components into an executable dataflow application including removing the design time links between ports. (Brief 8-9.) However, as already noted, Swamy discloses the compilation of the data transformation mapping into an executable dataflow application, and frees the generated node dependencies memory (associated with the design time links between the ports) after code generation is complete. See, e.g., Swamy at col. 2, lines 60-64; col. 13, lines 5-14.

Because the system and methods disclosed by Swamy meet all of the limitations as recited in claims 1, 2, 4-14, and 16, the rejection of claims 1, 2, 4-14, and 16 under § 102(e) should be sustained.

Rejection of claims 3 and 15 under 35 U.S.C. § 103(a) (Brief 9-10)

Claims 3 and 15 quite broadly recite "encrypting the dataflow graphs prior to the step of compiling the map component," Claim 3, and "some of the components being encrypted and comprising a proprietary data transformation," Claim 15. Appellants argued distinction of using the encryption "to protect the intellectual property in the component design" is simply not in the claims. (Brief 9.)

The portions of Block cited by the examiner in the rejection of claims 3 and 15 (which appellants conveniently leave out of their quote on p. 10 of the Brief) describe data transformation applications where input data is transformed and mapped to output data, the same relevant field of endeavor as the Swamy reference. Block col. 6, line 44, through col. 7, line 4. Block additionally teaches that it is known and beneficial to encrypt the data as part of the proprietary data transformation process. Block at col. 7, lines 9-13.

Because the combined teachings of Swamy and Block show the use of known techniques serving the same purpose of providing a data transformation application, and the combination of such known components being used merely as they were intended to use would yield predictable results (i.e., the transformation of data with the known benefits of encryption), the rejection of claims 3 and 15 under § 103(a) should be sustained.

Rejection of claims 17-20 under 35 U.S.C. § 103(a) (Brief 10-12)

As discussed in detail above regarding the rejections under § 102(e), Swamy discloses compiling user-generated mappings comprising data transformations into executable dataflow applications. See, e.g., Swamy at col. 2, lines 60-64 ("The invention takes a user-defined mapping between a source schema representing the definition of a source document, and a target schema representing a target document, and compiles the mapping into code for execution in a computer system.").

Appellants argue that, "there is no suggestion that dividing data and processes and pipelining in parallel would be applicable to Swamy." (Brief 11.) The examiner disagrees. The cited portions of Colliat teach that in implementing data transformation applications (like that disclosed in Swamy), "the optimization of the interface is essential, so that it can run within the narrow time frames required by business." Colliat at col. 2, lines 47-49. To achieve such an essential goal, Colliat teaches that, "Paralleling as much of the processing as possible can optimize the interface." *Id.* at col. 2, lines 53-54. Colliat then describes two different ways that such data transformation applications can be parallelized, including partitioning the data that need to be transformed and transforming all the partitions in parallel threads. *Id.* at col. 2, lines 56-57.

Because the combined teachings of Swamy and Colliat show the use of known techniques serving the same purpose of providing a data transformation application, and the combination of such known components being used merely as they were intended to use would yield predictable results (i.e., the transformation of data with the known benefits of increased

efficiency by running multiple data transformations on parallel threads), the rejection of claims 3 and 15 under § 103(a) should be sustained.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/Eric B. Kiss/
Eric B. Kiss
Primary Examiner, Art Unit 2192

Conferees:

/Tuan Q. Dam/
Tuan Q. Dam
Supervisory Patent Examiner, Art Unit 2192

/Lewis A. Bullock, Jr./
Lewis A. Bullock, Jr.
Supervisory Patent Examiner, Art Unit 2193